Alternative methods to extract CN signatures

Lena Morrill Gavarró

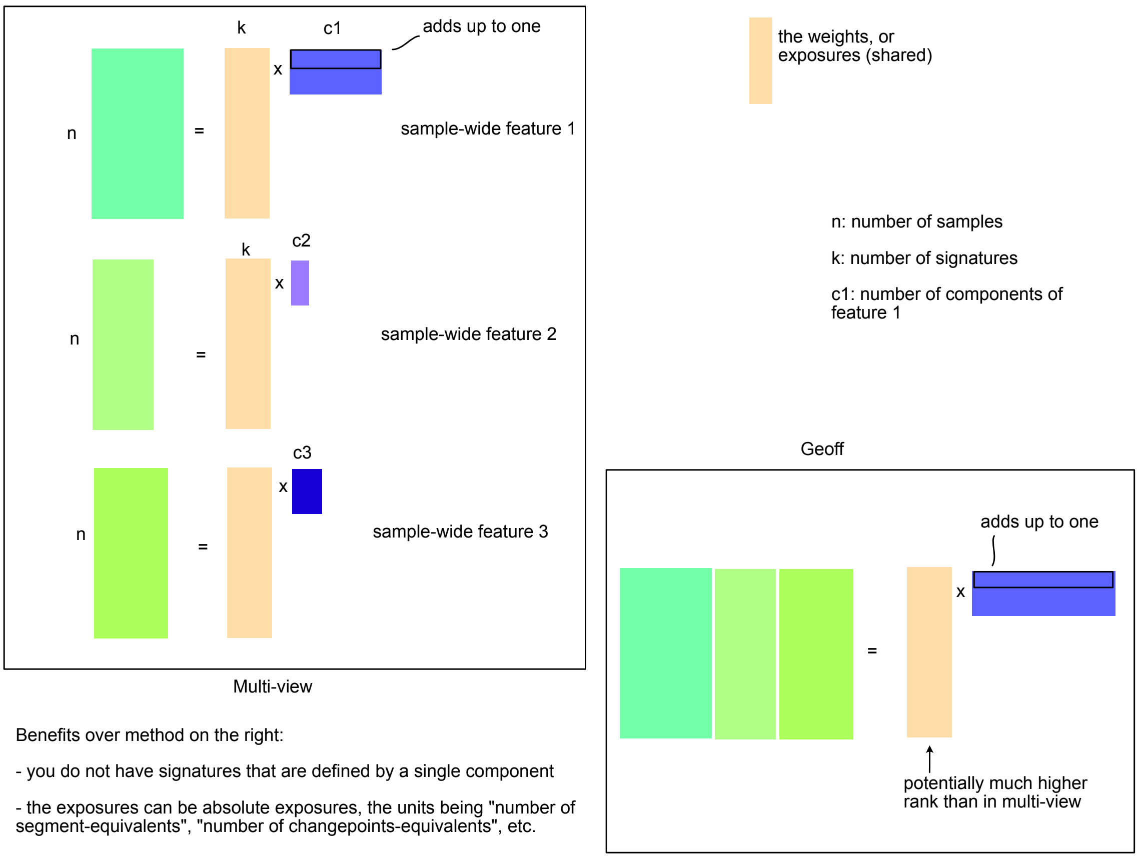
October 15, 2021

Following the rejection from Nature and comments from reviewer #3, September 2021.

Contents

1 Alternatives to NMF as it is

- Multi-view NMF (features and sum of posters matrix would stay as it is). See Multi-View Clustering via Joint Nonnegative Matrix Factorization, or (and I think it's the same?) integrative NMF.
- LDA (but then we need to redefine the features). What Jorge did.



Benefits over method on the right:

- you do not have signatures that are defined by a single component

- the exposures can be absolute exposures, the units being "number of segment-equivalents", "number of changepoints-equivalents", etc.

Figure 1: Current NMF and integrative NMF

2 The simulation from the reviewer

There is something slightly wrong in the simulation in that sample 1, 2, etc. have values for feature1 in multiple components, but for feature 2 they are all either 50 or 100, explaining partially why the signatures recovered are different. *Update after meeting 20210916: it's not that the simulation is wrong, but that the number of features that they use (number of CNAs) only has one observation per sample! So the simulation differs from our data in that we don't have any feature for which we only have one observation per sample.*

However, despite this apparent problem with their simulation

1. It's true that there is nothing stopping NMF from putting a lot of weight in a feature and very little in another
2. Re-normalising the signatures doesn't solve the problem.
3. A straightforward way of preventing this from happening is, as the reviewer says, to use joint (or multi-view) NMF
4. In my simulations (in which I was trying to recreate the problem that the reviewer refers to) the weights of both features in the signature definitions have been pretty much 0.5 - 0.5 when we use the true number of signatures. See section below.

2.1 My simulations

At least in some cases (probably provided the number of estimated signatures is the number of simulated signatures, and we have enough data i.e. segments per sample), the 0.5-0.5 split in probabilities for each of the two features is preserved in the estimates (this is desirable). But, very easily, we can get signatures which have a weight in a feature and no weight in the other (see example of signature 4 the right in Figure 2, which is what you call *ill-defined* in the review. Ill-defined has another meaning in statistics so, moving forward, I would call it something else.

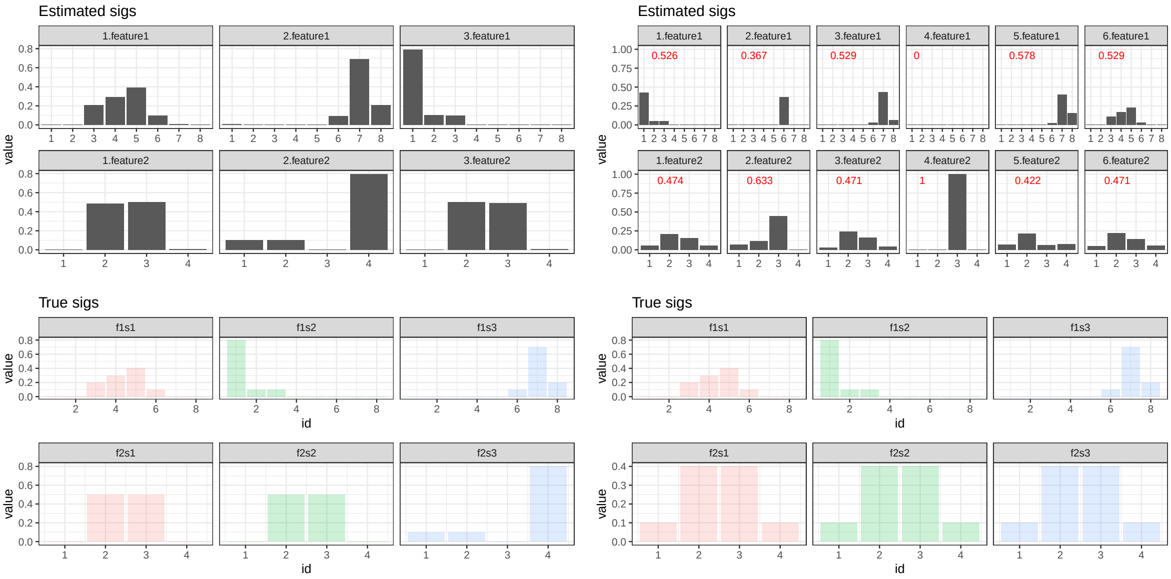


Figure 2: Satisfactory signature recovery (left) and unsatisfactory signature recovery (right)

Conclusion From my point of view, even though the simulation is not convincing enough to me, I agree with the point that the reviewer is making.

Question: Do we actually *want* signatures that have different weights for different features? e.g. a signature which is just e.g. "noise from the interaction between two types of signatures" that captures the extra segments that you get when you have two copy number events one on top of the other. Even if the answer is yes, then our way of analysing signatures individually is wrong, because the exposures no longer represent distinct processes of mutation.

3 IntNMF

In this section we assume we can solve the problem by just ensuring that each signature is defined by all features equally. I am using the R package IntNMF (see paper `https://pubmed.ncbi.nlm.nih.gov/28459819/`).

I am using the original features/sum of posterior matrix. As there are two features with three components, the maximum rank (i.e. number of signatures) is 2. Removing these two features (bp10MB, osCN) with two components, we can have as many as 5 signatures.

3.1 Normalisation (or not) of the sum-of-posteriors matrix

Note: this section is technical and not too interesting

Normalising the values of components within a feature does not give the same results as not normalising them, even though intuitively I thought it would. Using this singular-covariance input matrix (which we get if we normalise it) is fine for the IntNMF algorithm. Using the normalised features seem to give better results, in that there are fewer extreme values/outliers in the exposures.

In the output from IntNMF

- Exposures are not normalised sample-wise (as expected)
- Weights are not normalised feature-wise (not as expected)
- The sum of weights of signatures across features is not even constant (this is what I find most baffling) even though it's quite similar

3.1.1 Getting normalised signatures

1. Run IntNMF
2. Compute the average sum of weights for each signature, across features. This gives a vector a
3. Multiply the extracted exposures matrix W by $(diag(a))^{-1}$, i.e. $\bar{W} = W \cdot (diag(a))^{-1}$
4. Normalise $\bar{W}$ row-wise (i.e. sample-wise) to get a normalised exposures matrix

3.2 Comparison to NatGen signatures

Input matrix type	Exposures used	k	Comparison to NatGen
Normalised feature-wise, removing 2 features	Raw, non-normalised	3	s2 and s3 have equivalents. s4 to a lesser extent
Normalised feature-wise, removing 2 features	Raw, non-normalised	4	s2, s3 s4 and s5 have equivalent signatures in the new ones
Normalised feature-wise, removing 2 features	Raw, non-normalised	5	The same; s2, s3 s4 and s5 have equivalent signatures in the new ones
Normalised feature-wise, removing 2 features	Normalised with diagonal matrix as above	5	S1, s2, s3 s4 and s5/s7 have equivalent signatures in the new ones

Table 1: Comparison of new and previous signatures

Most signatures have pretty clear equivalents, with s6 and s7 being exceptions. The two groups defined by (for the most part) s3 and s4 are still recovered even when we only use 2 signatures (see Figure 3).

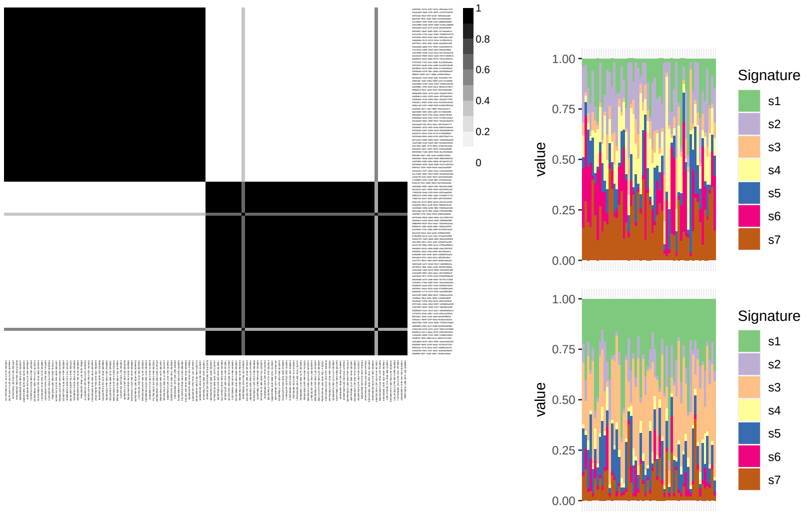


Figure 3: Using all features, with components normalised feature-wise, with rank $k = 2$. The two groups represent the split of s3/s4 in the NatGen signatures.

$*$ *To do, suggested by Ruben 20210916: compare signature definitions (although then I have to think whether I need to re-normalise signature definitions per feature or not).*

$*$ *20210916: by "ill-defined", in the review+rebuttal, we mean signatures which look weird because e.g. they don't have segment size*

$*$ *20210916: the difference in s3 and s7 is not due to brca/non-brca. s7 is when there is WGD, and s3 is on a diploids genome. If we remove the copy number feature and s3 and s7 should be extracted as a single signature*

3.2.1 Comparison of exposures

Figure 4 shows the correlation of exposures of original and re-derived signatures.

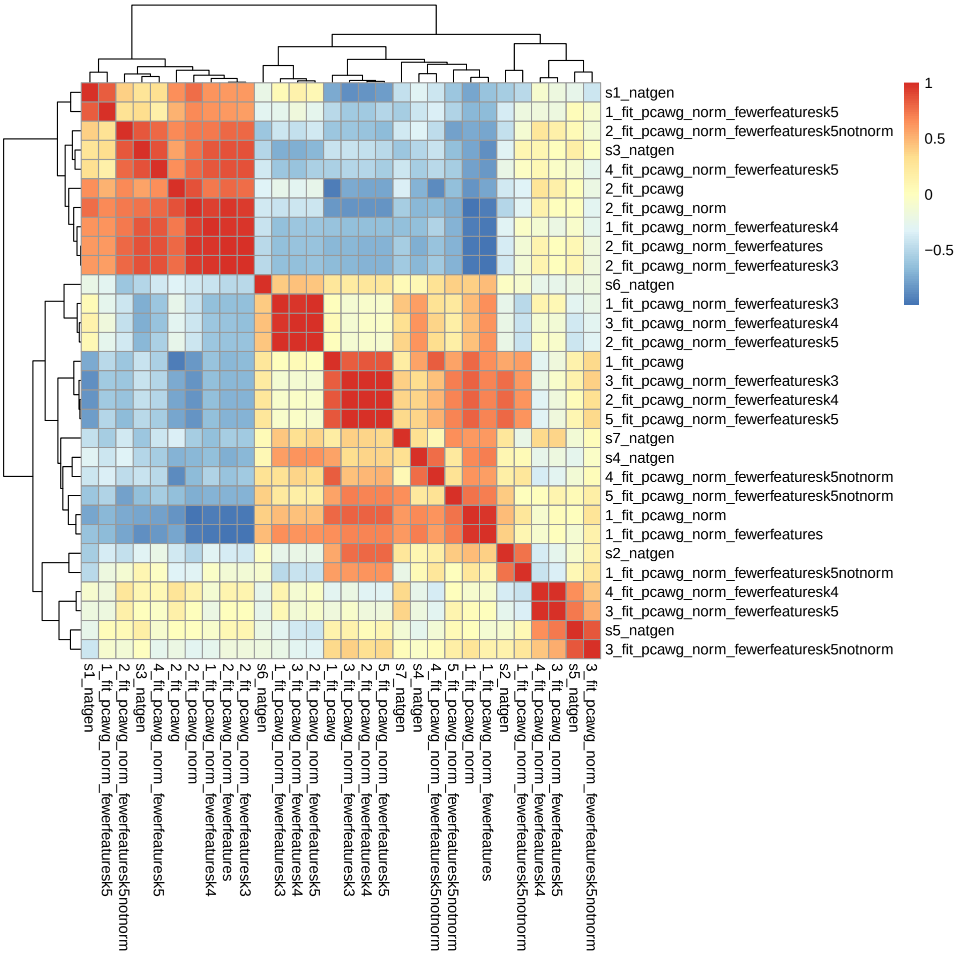


Figure 4: Clustering of exposures with original and re-derived signatures

- s1: there is an equivalent only in `1_fit_pcawg_norm_fewerfeaturesk5` (i.e. when I haven't normalised the features to create the input matrix, and I haven't re-scaled the output of the NMF), but not to any `*_fit_pcawg_norm_fewerfeaturesk5notnorm`.
- s2: similar to `1_fit_pcawg_norm_fewerfeaturesk5notnorm`
- s3: with exposures very similar to those of many other signatures, e.g. `4_fit_pcawg_norm_fewerfeaturesk5`
- s4: similar to `4_fit_pcawg_norm_fewerfeaturesk5notnorm`
- s5: similar to `3_fit_pcawg_norm_fewerfeaturesk5notnorm`
- s6: it doesn't haven exposures similar to any other signature (but, in fairness, we only have 5 signatures!)

- s7: it doesn't haven exposures similar to any other signature (again, we only have 5 signatures)

3.2.2 Comparison of signature definitions

Analysis per feature

- segsize: s2, s4 and s5 nearly identical, and similar to many new signatures. s7 and s3 very similar, and very similar to `5_fit_pcawg_norm_fewerfeaturesk5notnorm` and other signatures. s1 similar to plenty of other new signatures, same for s6.
- bp10MB: s2, s4 and s5 very similar. s3 and s7 very similar. s1 has an equivalent in `2_fit_pcawg_norm` *(note: this is a feature that is not used when only using 4 features!)*
- osCN: s4, s6, s5 and s7 very similar. s3 very different. s1 and s2 very similar, and to `2_fit_pcawg_norm` *(note: this is a feature that is not used when only using 4 features!)*
- changepoint: s4 is different to anything else. s6 is similar to many signatures. S5 has many perfect correlates. S1 doesn't have any equivalent. s2, s3 and s7 are all similar to each other and other signatures.
- Copy number: s6 and s4 don't really have true equivalents. s5 has a perfect one in `3_fit_pcawg_norm_fewerfeaturesk5notnorm`. s2 and s7 only correlated (heavily) with each other. S1 doesn't have any very clear equivalent.
- bpchrarm: s4 has the equivalent in `3_fit_pcawg_norm_fewerfeaturesk5notnorm`, s2 in `4_fit_pcawg_norm_fewerfeaturesk5notnorm`. S5 doesn't have clear equivalents but does correlated heavily with plenty of other signatures. s6 is equivalent to `5_fit_pcawg_norm_fewerfeaturesk5notnorm` and, to a lesser extent, to s3 and s7 (which are nearly-identical but don't have equivalents in `*_fit_pcawg_norm_fewerfeaturesk5notnorm`, only in other types of extracted signatures). S1 correlates very well with many other signatures, including one of the family above: `2_fit_pcawg_norm_fewerfeaturesk5notnorm`.

Figure 5 shows the overall correlation, when including all features. The definitions of the original signatures have been re-normalised feature-wise. The equivalences are

Natgen signature	Re-derived signature	Data supporting equivalence
s1	`2_fit_pcawg_norm_fewerfeaturesk5notnorm`	Overall signature definition
s2	`1_fit_pcawg_norm_fewerfeaturesk5notnorm`	Overall signature definition
s3	none	
s4	`4_fit_pcawg_norm_fewerfeaturesk5notnorm`	Overall signature definition
s5	`3_fit_pcawg_norm_fewerfeaturesk5notnorm`	Overall signature definition
s7	`5_fit_pcawg_norm_fewerfeaturesk5notnorm`	Overall signature definition
s6	none	

Table 2: Equivalences when it comes to signature definitions

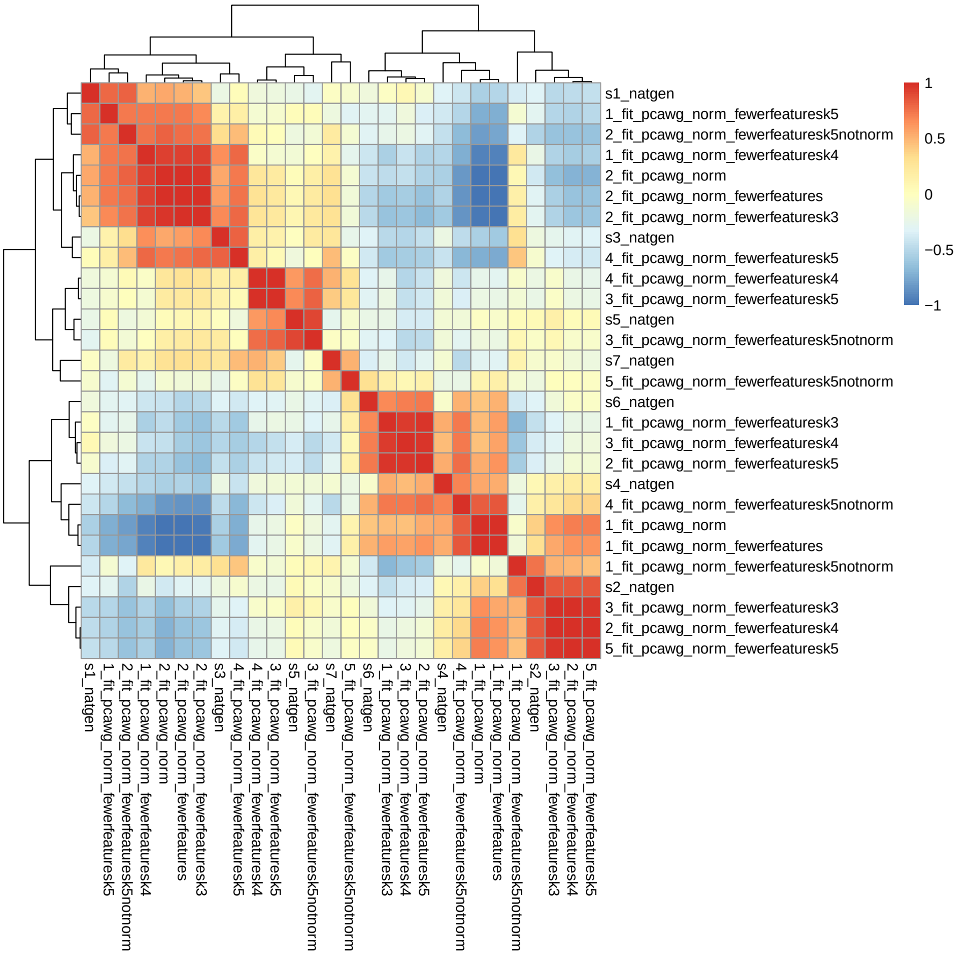


Figure 5: Clustering of signature definitions of original and re-derived signatures

3.2.3 Conclusion

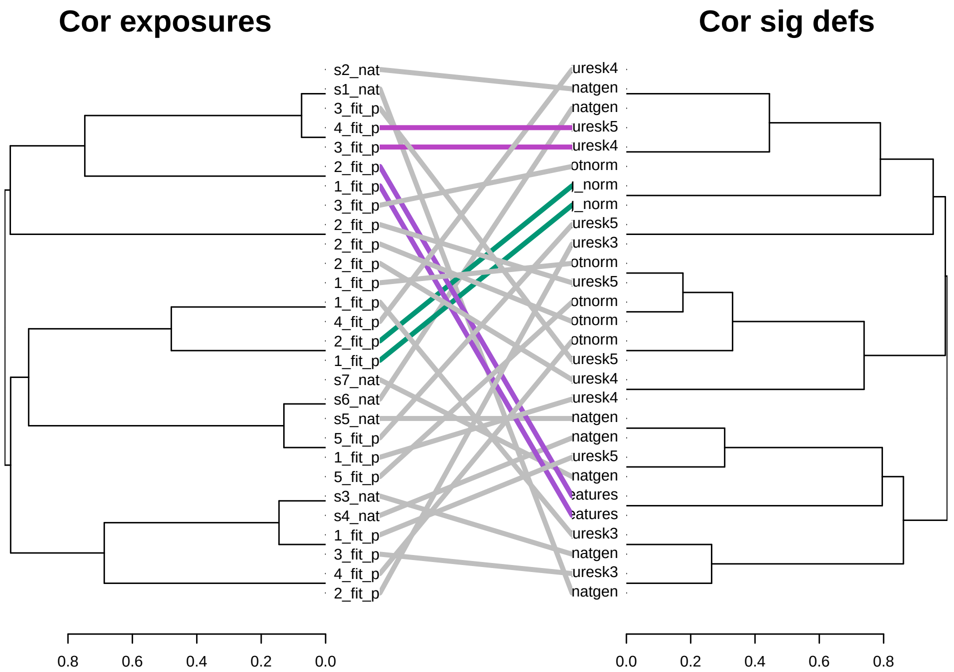


Figure 6: Tanglegram of the clustering of signature definitions and of exposures, for original and re-derived signatures

4 Simulation of derivative chromosomes for signature validation

4.1 Software for CN simulation

- SimSvGenomes (cpp programme): output sort of difficult to understand. It doesn't simulate genomes per se, rather, it outputs a bunch of segments. It doesn't simulate a single genome, but gives you the whole history (without explaining which genome comes from which). I gave up using it

- RSVSim: R package. What I am using right now. It creates deletions, insertions, etc. in a genome (either a true one or a simulated one)

- SECNVs: I haven't used it

- SCNVSim: I haven't used it

4.2 Pipeline for CN simulation

1. Create genome. Create a `BSGenome` R package with the genome
2. Create derivative genome, given some exposures (although right now I am just using random changes in the genome), using `RSVSim`.
   (a) Create reads from the derivative genome
3. Align the reads to the genome using bwa
4. Segment the genome using QDNASeq
5. Call copy number signatures

20210927 current problem: the estimated copy number is huge. Problem with sequencing depth? because I am estimating it from the average number of reads per position

5 My thoughts on the reviewers and rebuttal

- *However, their simulation suffers from a fatal design flaw and is not a valid representation of our approach. Given that reviewer 3's harsh criticism was based on a flawed argument, we would like to discuss whether our manuscript could be reconsidered at Nature.* Despite the flaw in their simulation, the point they are making is valid. Also keep in mind - it's an example.
- It's true that as the number of within-patient observation decreases (e.g. if we had a feature which is genome-wide), we would encounter exactly the same situation that the reviewer refers to. We would have a single observation categorised in one of multiple components (multiple because we are pooling all observations from all samples when we are computing the sum of posteriors). Right now the feature with the least number of observations is the number of breakpoints per chromosome arm, where we have 44 observations per sample (if we have 22 autosomes; fewer if we have lost any). However, we also use a sum of posteriors, instead of the MAP, making it easier to have the signal spread (but not enforcing it).
- In Ruben's simulation, we can reconstruct the signatures where but that is because **we are using the correct number of signatures**. If we use more signatures (see my simulation, above) we have these badly-sparse signatures.
- (from markdown document) *they encode the number of copy number events as an ordinal feature (categories with an order) not a count based feature* I don't see how that's true. They have categories, you have categories. I don't think the sentence above makes sense
  1. The fact that the reviewer is using ordinal categories (which we are too!) does not affect anything
  2. *However, none of the samples actually represent one of the three signatures as no sample has weights on both components of the number of CNAs feature as defined above.* We are missing the point - the point is that there is a single observation

in the reviewer's simulation, because he is counting the number of CNAs in the sample!

3. *In our method we always use count based feature encodings and never categorical.* what does that mean

4. *In this case, we can keep the changepoint weights uniformly distributed across the features, however, as the breakpoints per chromosome arm feature has a finite space (only 44 autosome arms) in which to dribute the breaks, we cannot have a uniform distribution, but rather one which has stronger weight on the zero component.* why. What does that mean. You mean that the sum of posteriors in this feature is lower than the sum of posterior in the other feature, and hence that there is inherently in the method a lower weight in this feature (which is true)?

6 Simulating genomes - 20211014

1. the reads are not uniform. currently simulating many reads to see if it's simply a problem of number of reads. With more reads it's better, but still not perfect uniform coverage at all.

2. even with many reads, and with a single deletion of 1000bp, there is no way of noticing a difference in the reads

3. Now, manually, removing the first 1kb of a chromosome (both chromosomes). From inspecting the bam files, it looks fine.

   (a) reducing the number of reads gives a reconstruction which is really not as good

   Parameters:

   (a) Number of reads
   (b) Length of reads
   (c) Length of chromosomes
   (d) Length of bins for QDNAseq
   (e) Purity
   (f) CN aberrations which are simulated (how many copies, etc.)

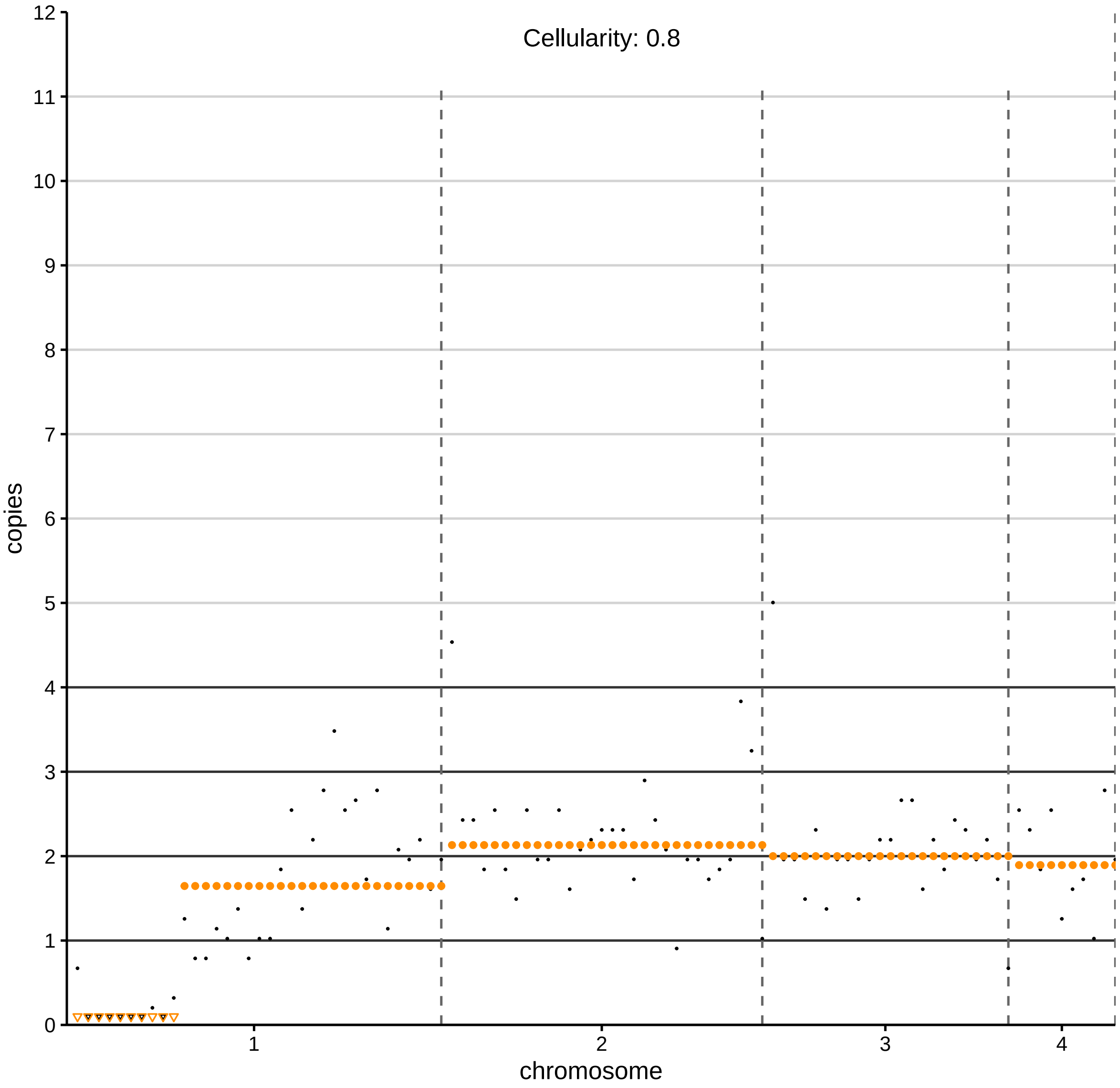


Figure 7: Notice the first part of chrom 1 in which there is a deletion of both alleles